Explicaciones Dialécticas

Nicolás D. Rotstein Alejandro J. García Guillermo R. Simari

e-mail: {ndr,ajg,grs}@cs.uns.edu.ar
Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
Laboratorio de Investigación y Desarrollo en Inteligencia Artificial
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Av. Alem 1253, (8000) Bahía Blanca, Argentina

1. Introducción

Este artículo reporta el estudio realizado hasta el momento en la línea de investigación de *explicaciones dialécticas* [4] y propone direcciones para el trabajo a futuro. Dentro de varias áreas de la Inteligencia Artificial se ha puesto atención al rol de las explicaciones, en particular en sistemas expertos [7, 10, 6]. El objetivo de brindar explicaciones en sistemas expertos es brindar mayor confianza al usuario con respecto a las respuestas dadas por el sistema. No obstante, pocos han analizado el uso de explicaciones dentro de sistemas argumentativos [8]. En general, se habla de 'argumento' como una explicación para un literal dado; es decir, el punto siendo explicado es puesto en discusión, y a partir de ahí puede ser aceptado o no. Por ejemplo, el argumento "*hoy está lloviendo, por lo cual me voy a quedar a dormir*" puede ser una explicación para el literal "*hoy no voy a trabajar*". Sin embargo, la razón provista para no ir a trabajar puede ser puesta en tela de juicio. Por otro lado, en revisión de creencias también se ha estudiado el rol de las explicaciones [3]. Éstas son utilizadas para resolver inconsistencias entre las creencias y las nuevas percepciones; aquel literal soportado por la explicación más *fuerte* es el que finalmente prevalecerá.

Si bien reconocemos que un argumento en sí puede ser tomado como una explicación para un literal, en esta línea de trabajo estamos enfocados en otro tipo de explicaciones: aquellas que justifican las *garantías* brindadas por un sistema argumentativo. Nuestro trabajo puede analizarse desde un punto de vista abstracto, y los resultados se implementarán en un formalismo en particular: Defeasible Logic Programming (DeLP) [5]. Este formalismo utiliza programas lógicos rebatibles (de aquí en adelante, *DeLP-programs*) para representar el conocimiento, y argumentación rebatible para el razonamiento. El procedimiento de prueba utilizado es *dialéctico*, por lo cual estudiaremos un concepto al que denominaremos *explicaciones dialécticas* (de aquí en adelante, δ*-Explanations*). El procedimiento de prueba, ante una consulta (*query*), responde si puede o no creerse en dicho literal, es decir, si éste puede considerarse *garantizado*. El mecanismo para lograrlo es construir árboles de dialéctica enraizados en un argumento para el literal consultado o para su complemento. Para más detalles referirse a [5]. El conjunto completo de árboles de dialéctica relacionados con la consulta realizada será considerado la δ*-Explanation* de la respuesta para esa consulta.

A lo largo del artículo mostraremos cómo las δ*-Explanations* son una herramienta poderosa para comprender y analizar las interacciones entre argumentos dentro de un árbol de dialéctica, y para asistir la codificación y el depurado de bases de conocimiento. Actualmente, hay un prototipo implementado en nuestro laboratorio que permite visualizar el conjunto de árboles de dialéctica que

justifican la respuesta dada para una consulta. Los ejemplos mostrados en este artículo están generados a partir de este prototipo.

En esta línea de investigación se propone el estudio de un área poco explorada: la de explicaciones en sistemas argumentativos. Las explicaciones que proveemos apuntan a revelar el contexto completo de una consulta. Los ejemplos dados en este artículo enfatizan este punto. En la sección que sigue describimos y ejemplificamos las explicaciones dialécticas tanto para consultas fijas como con variables, en la sección 3 discutimos y hacemos una comparación con un acercamiento que se relaciona al nuestro, y en la sección 4 delineamos el trabajo futuro sobre esta línea.

2. Explicaciones Dialécticas

Una *consulta DeLP* (*DeLP-query*) es un literal fijo que DeLP intentará garantizar. Una consulta con al menos una variable será llamada *consulta esquemática* (*schematic query*) y representará el conjunto de *DeLP-queries* que unifican con ella. Ambos tipos de consultas requieren tratamientos diferentes, por lo cual primero ejemplificaremos las explicaciones para *DeLP-queries*, y luego, para *schematic queries*.

Pueden existir varios argumentos para un mismo literal, y cada argumento generará un árbol de dialéctica distinto. Por esto, la respuesta devuelta ante una consulta es sólo la 'punta del iceberg' de un proceso que incluye la exploración del conjunto de árboles de dialéctica que soporta dicha respuesta. Por lo tanto, para entender realmente por qué una consulta tiene una respuesta en particular, es esencial considerar qué argumentos han sido generados y qué conexiones existen entre ellos. Una δ-*Explanation* para una *DeLP-query* Q es la unión de los árboles de dialéctica construibles a partir de cada argumento para Q y para el complemento de Q.

Ejemplo 1 (DeLP-query) *Consideremos un programa lógico rebatible* (Π_1, Δ_1)*:*

$$\Pi_1 = \{q, t\} \quad \Delta_1 = \left\{ \begin{array}{ll} (r \prec q), & (\sim r \prec q, s), \\ (r \prec s), & (\sim r \prec t) \end{array} \right\}$$

desde el cual se pueden construir los siguientes argumentos:

- $\langle \mathcal{R}_1, \sim r \rangle$ = $\langle \{\sim r \prec t\}, \sim r \rangle$
- $\langle \mathcal{R}_2, r \rangle$ = $\langle \{r \prec q\}, r \rangle$

A partir de este programa la respuesta para la DeLP-query r *es* UNDECIDED*, cuya* δ-Explanation *puede verse en la Figura 1. Allí puede verse que ambos argumentos están en una situación de bloqueo; es decir, ninguno bloquea al otro.*

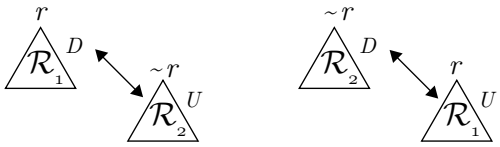

Figura 1: δ-*Explanation* para la consulta r del Ejemplo 1

A continuación, el Ejemplo 2 muestra cómo la introducción de un único hecho en el programa del Ejemplo 1 no sólo modifica la respuesta para la consulta r, sino que hace una diferencia significativa en la explicación. Es en estos casos en los que recurrir a las δ-*Explanations* se torna necesario para el completo entendimiento de lo que ocurre dentro del sistema.

Ejemplo 2 (Extiende al Ej. 1) *Consideremos el DeLP-program* $(\Pi_1 \cup \{s\}, \Delta_1)$. *Si realizamos nuevamente una* DeLP-query *por* r*, obtenemos* NO *como respuesta, y la* δ-Explanation *es la mostrada en la Figura 2. Esta explicación tiene dos árboles más que la mostrada en el ejemplo anterior, debido a dos nuevos argumentos que pueden extraerse del programa:*

- $\langle \mathcal{R}_3, r\rangle$ = $\langle \{r \multimap\!\prec s\}, r\rangle$
- $\langle \mathcal{R}_4, \sim r\rangle$ = $\langle \{\sim r \multimap\!\prec q, s\}, \sim r\rangle$

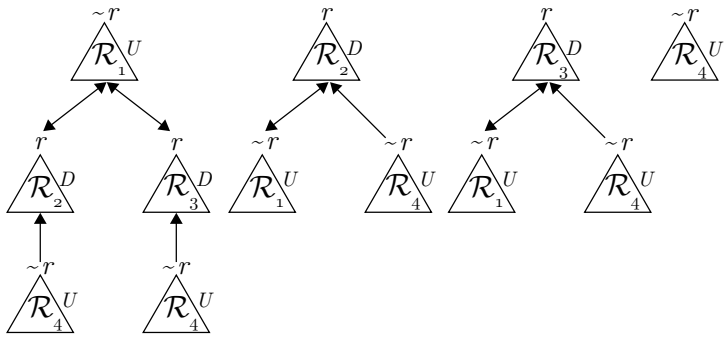

Figura 2: δ-*Explanation* para el Ejemplo 2

A diferencia de las explicaciones dialécticas para *DeLP-queries*, las δ-*Explanations* para las *schematic queries* tienen que lidiar con las diferentes instanciaciones posibles de la consulta dada. Por esto, una δ-*Explanation generalizada* para una consulta esquemática Q es la unión de todas las δ-*Explanations* para cada instancia de Q. A continuación, vamos a ilustrar este concepto con un ejemplo.

Ejemplo 3 *Consideremos el siguiente DeLP-program:*

$$\Pi = \left\{ \begin{array}{ll} (bird(X) \leftarrow chicken(X)) & chicken(little) \\ chicken(tina) & bird(rob) \\ scared(tina) & \end{array} \right\}$$

$$\Delta = \left\{ \begin{array}{l} flies(X) \multimap\!\prec bird(X) \\ flies(X) \multimap\!\prec chicken(X), scared(X) \\ \sim flies(X) \multimap\!\prec chicken(X) \end{array} \right\}$$

Supongamos que queremos saber, a partir de este programa, si se puede garantizar que algún individuo no vuela. Al consultar por $\sim flies(X)$ *la respuesta es* YES*, debido a que hay una instancia garantizada para esta consulta:* $\sim flies(little)$*. El argumento que soporta esta garantía es:* $\langle \mathcal{B}_2, \sim flies(little)\rangle$ = $\langle \{\sim flies(little) \multimap\!\prec chicken(little)\}, \sim flies(little)\rangle$*, el cual derrota a:* $\langle \mathcal{B}_1, flies(little)\rangle$ = $\langle \{flies(little) \multimap\!\prec bird(little)\}, flies(little)\rangle$*, por ser más directo.*

Los árboles de dialéctica que componen la explicación generalizada se muestran en la Figura 3. Esta explicación también muestra que la instancia $\sim flies(tina)$ *no está garantizada.*

Por otro lado, veamos que la respuesta para la schematic query *opuesta (i.e.,* $flies(X)$*) es también* YES*. Por supuesto, el conjunto de instancias garantizadas es distinto esta vez:* $flies(tina)$ *y* $flies(rob)$*. Los argumentos que soportan estas garantías pueden verse en la Figura 3, y son:*

- $\langle \mathcal{A}_1, flies(tina)\rangle$ = $\langle \{flies(tina) \multimap\!\prec bird(tina)\}, flies(tina)\rangle$
- $\langle \mathcal{A}_2, \sim flies(tina)\rangle$ = $\langle \{\sim flies(tina) \multimap\!\prec chicken(tina)\}, \sim flies(tina)\rangle$
- $\langle \mathcal{A}_3, flies(tina)\rangle$ = $\langle \{flies(tina) \multimap\!\prec chicken(tina), scared(tina)\}, flies(tina)\rangle$
- $\langle \mathcal{C}_1, flies(rob)\rangle$ = $\langle \{flies(rob) \multimap\!\prec bird(rob)\}, flies(rob)\rangle$

La δ-Explanation *generalizada para* $flies(X)$ *es la misma que aquella para* $\sim flies(X)$*, dado que las explicaciones para un literal también incluyen el análisis para el complemento del mismo.*

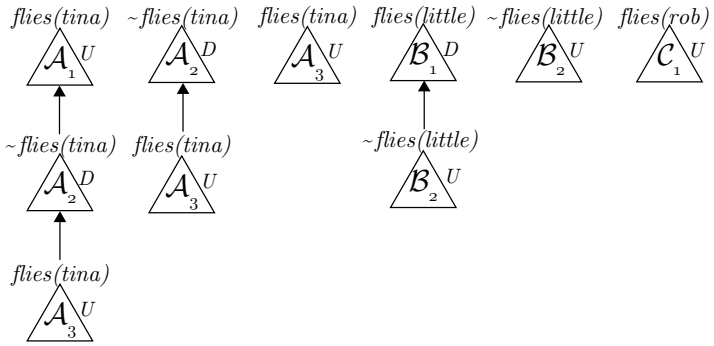

Figura 3: δ-*Explanation* generalizada para $\sim flies(X)$

Las consultas esquemáticas nos dan la posibilidad de hacer preguntas más generales que las *DeLP-queries*. Con ellas no estamos preguntando si se puede creer en una dada pieza de conocimiento, sino que estamos preguntando si existe una instancia de la consulta (relacionada a un individuo) que puede ser garantizada por el sistema. Esto puede llevar a un razonamiento más profundo, ya que se puede plantear una consulta, obtener las instancias garantizadas y continuar con la *discusión* con esos individuos. Es decir, la discusión puede ser focalizada.

3. Trabajo Relacionado

En el sitio web del proyecto ASPIC [2] se presenta una demostración de un *engine* argumentativo basado en el formalismo de Gerard Vreeswijk. En ella se brindan varios casos de ejemplo, dando la posibilidad de ejecutar consultas sobre esas bases de conocimiento, e incluso editarlas. También se puede crear una base de conocimiento nueva, y realizar consultas sobre ella.

Es interesante la posibilidad de visualizar, mediante un grafo, el proceso argumentativo provocado por la consulta. Esto puede tomarse como una explicación para la respuesta dada. Los argumentos y las relaciones de ataque entre ellos se muestran de acuerdo al *Argument Interchange Format* (AIF) [1], y los nodos están adecuadamente coloreados para representar argumentos derrotados (rojo) y no derrotados (verde). Sin embargo, la visualización resulta confusa, ya que hay demasiados elementos en pantalla. Existen estilos de flecha de distintos colores utilizados para representar *rebut*, *undercut*, *etc.* y pueden hacerse muy difíciles de seguir, sobre todo por su longitud. Además, las estructuras internas de los argumentos no son simples y se muestran desde un principio en completo detalle.

En cambio, la representación dada por nuestro sistema es estructuralmente más simple, ya que se trata de árboles, y los nodos (*i.e.*, argumentos) se muestran en una versión simplificada, con la posibilidad de ver su contenido mediante un *tooltip*. Uno o varios nodos pueden ser expandidos y contraídos para revelar la derivación lógica que hizo a la construcción del argumento. También se puede hacer *zoom in/out* (útil en caso de árboles muy grandes) y un mapa de referencia facilita la ubicación del foco de la ventana de visualización dentro de la pantalla.

Por otra parte, en [2] se brinda otra opción para obtener una explicación de la respuesta dada por el *engine*: una traza que muestra no sólo la construcción de argumentos, sino que también describe interferencias y defensa entre ellos y el status de cada uno: PRO o CON, es decir, a favor o en contra del argumento que soporta la consulta, respectivamente. Se detalla el grado de fortaleza de la interferencia, es decir, si un argumento logró derrotar a otro. No obstante, la traza tiene debilidades similares al grafo en cuanto a visualización: es compleja desde un principio, y no brinda interacción alguna, *i.e.*, el usuario no puede contraer/expandir secciones de la traza. El detalle con el que se muestra la traza parece un tanto excesivo, lo cual afecta a la usabilidad. Nuestro sistema no cuenta con este tipo de facilidades para trazar el proceso argumentativo.

4. Conclusiones y Trabajo Futuro

Esta nueva línea de trabajo en el área de argumentación está dedicada a estudiar una parte poco explorada: la de explicaciones en sistemas argumentativos, plasmando los resultados en el sistema DeLP. Este estudio abre nuevas posibilidades para implementar, por ejemplo, sistemas expertos basados en DeLP. La idea de este trabajo nació como una necesidad al hacer testeo y depurado de programas lógicos rebatibles, ya que se torna realmente difícil seguir la dinámica argumentativa del sistema.

Colateralmente, esta línea abarca el análisis e implementación de nuevos sistemas de visualización que faciliten la tarea del usuario en el uso de las explicaciones. Este tipo de herramientas se torna necesario debido a que el número y tamaño de los árboles de dialéctica generados por una consulta pueden llegar a ser muy incómodos de manejar. Por esto, contar con la posibilidad de visualizar los árboles con las facilidades que mencionamos a lo largo del artículo hace que las explicaciones dialécticas cobren mayor sentido. Actualmente, existe un prototipo del sistema DeLP que ofrece una visualización interactiva de explicaciones dialécticas. Como próximo paso de investigación, nos dirigimos a la inclusión de nueva información dentro de las explicaciones.

Referencias

[1] Carlos Chesnevar, Jarred McGinnis, Sanjay Modgil, Iyad Rahwan, Chris Reed, Guillermo Simari, Matthew South, Gerard Vreeswijk, and Steven Willmott. Towards an argument interchange format. *Knowl. Eng. Rev.*, 21(4):293–316, 2006.

[2] ASPIC Project Argumentation Engine Demo. http://aspic.acl.icnet.uk/ArgumentationSystem.

[3] Marcelo A. Falappa, Gabriele Kern-Isberner, and Guillermo R. Simari. Explanations, belief revision and defeasible reasoning. *Artif. Intell.*, 141(1):1–28, 2002.

[4] A. Garcia, C. Chesnevar, N. Rotstein, and G. Simari. An abstract presentation of dialectical explanations in defeasible argumentation. *Workshop in Argumentation and Non-Monotonic Reasoning (ArgNMR)*, 2007 (to appear).

[5] Alejandro J. García and Guillermo R. Simari. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.

[6] Giovanni Guida and Marina Zanella. Bridging the gap between users and complex decision support systems: the role of justification. In *ICECCS '97: Proc. 3rd IEEE International Conference on Engineering of Complex Computer Systems*, pages 229–238, Washington, DC, 1997.

[7] Carmen Lacave and Francisco J. Diez. A review of explanation methods for heuristic expert systems. *Knowl. Eng. Rev.*, 19(2):133–146, 2004.

[8] B. Moulin, H. Irandoust, M. Bélanger, and G. Desbordes. Explanation and argumentation capabilities: Towards the creation of more persuasive agents. *Artif. Intell. Rev.*, 17(3):169–222, 2002.

[9] Michael Schroeder. Towards a visualization of arguing agents. *Future Generation Computer System*, 17(1):15–26, 2000.

[10] L. Richard Ye and Paul E. Johnson. The impact of explanation facilities on user acceptance of expert systems advice. *MIS Q.*, 19(2):157–172, 1995.